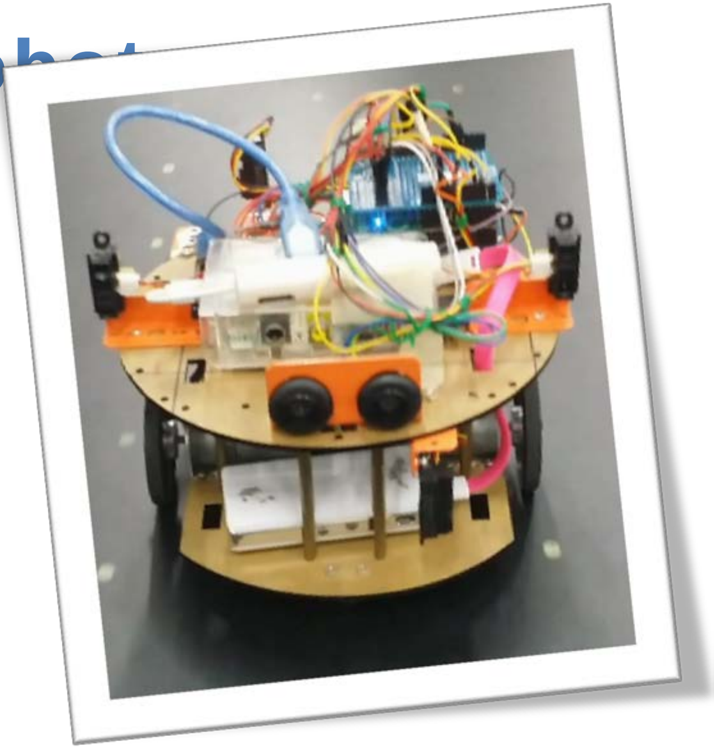
NANYANG TECHNOLOGICAL UNIVERSITY

MDP: Sensor Interfacing and PID Controller for 2-wheel drive robot



presented by

Suresh Sundaram
Associate Professor
School of Computer Engineering

Sept. 2018

Overview

Open Loop Control

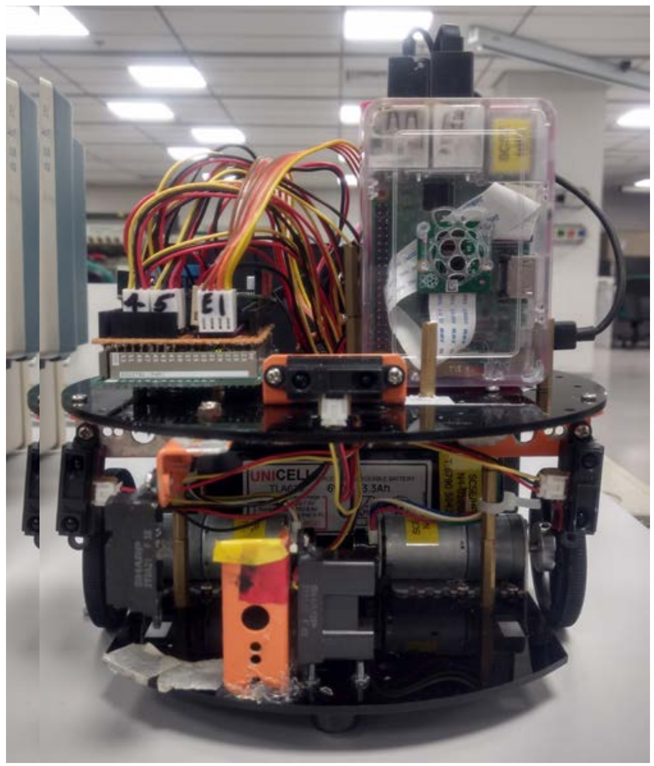
PID Control

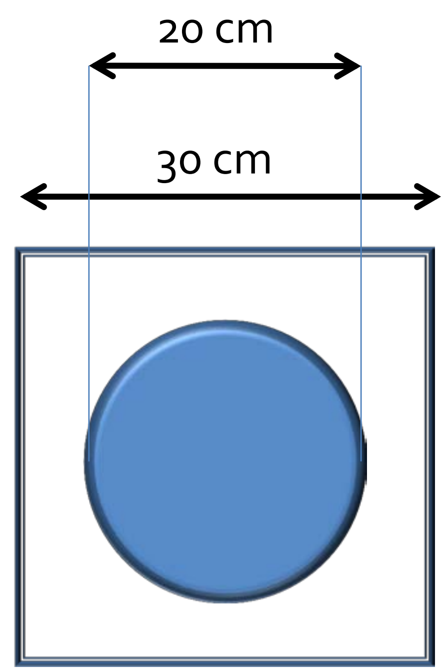
2 WD Speed Control

Issues

Overview of Assembled Robot

Fully Assembled System





Robot Footprint
in the Maze

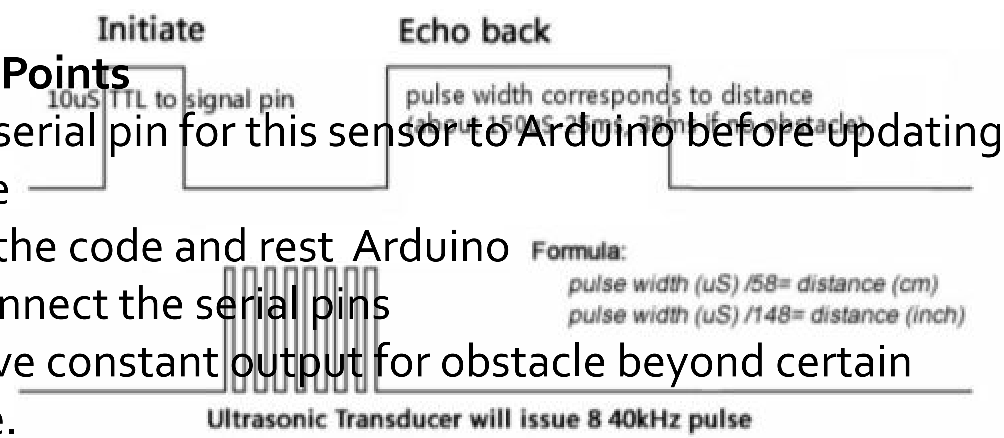
Overview
Open Loop Control
PID Control
2 WD Speed Control
Issues

Understanding of Ultrasonic Sensor

Specification of URM37
Power : 5V and 20mA
Beam width : 60 Deg
Detecting range: 5cm-500cm
Resolution: 1cm (50 microsec)
Note: Measuring time is 100ms

Important Points

- Unplug serial pin for this sensor to Arduino before updating the code
- Upload the code and rest Arduino
- Then connect the serial pins
- It will give constant output for obstacle beyond certain distance.



Example Timing from HC-SR04

Understanding of IR Sensor

- This is a parallax (or triangulation) sensor.
- Emitter is IR LED
- Detector is IR Photodiode
- Resistance changes based on angle of received IR light

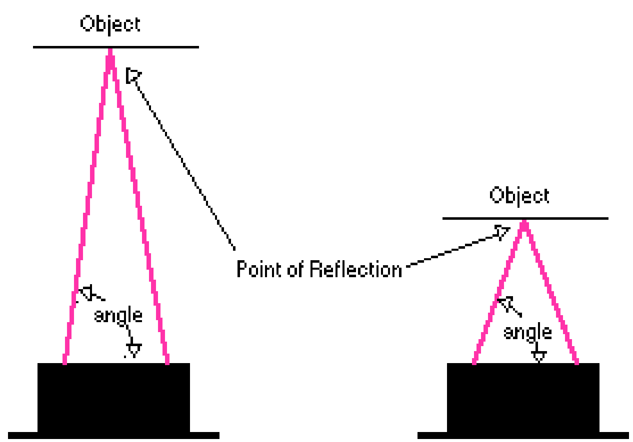
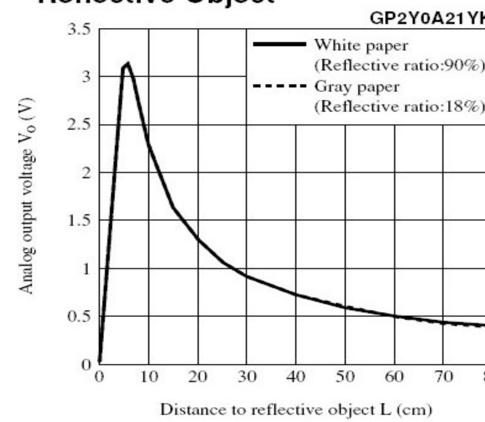
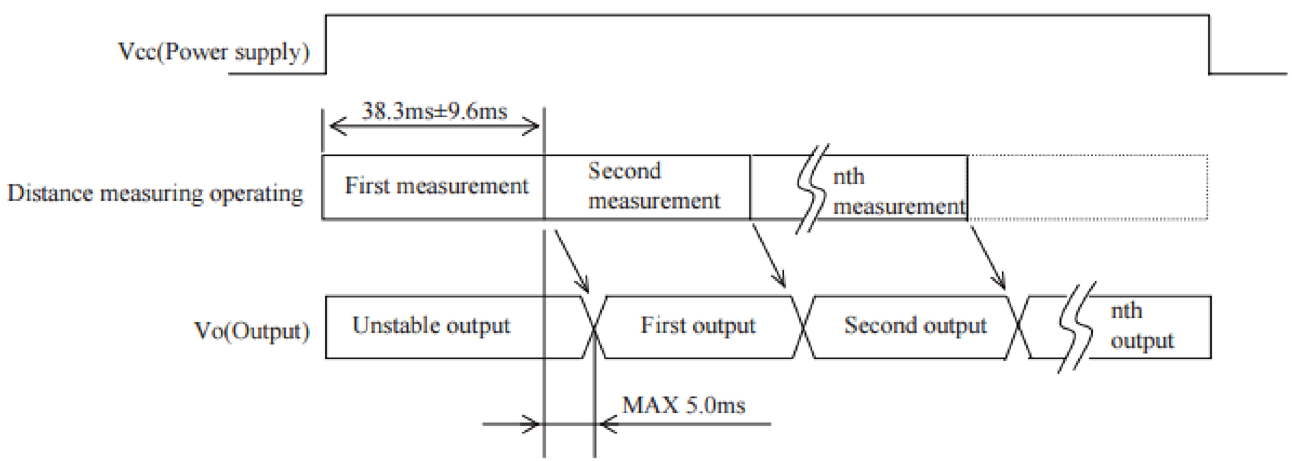


Fig.5 Analog Output Voltage vs. Distance to Reflective Object



Understanding of IR Sensor

- First measurement will be slow when compared to subsequent measurement. Ignore the first two measurements and implement median filter



Understanding of IR Sensor

- The accuracy/precision of sensor will decrease with distance

Real Distance in cm	Measured Distance in cm
10	12
20	21
30	33
40	44
50	56
60	68
70	79

Understanding of IR Sensor

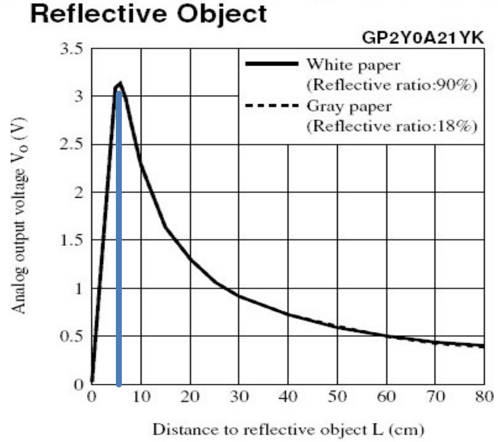
- Accuracy will be different for different angle of mounting.

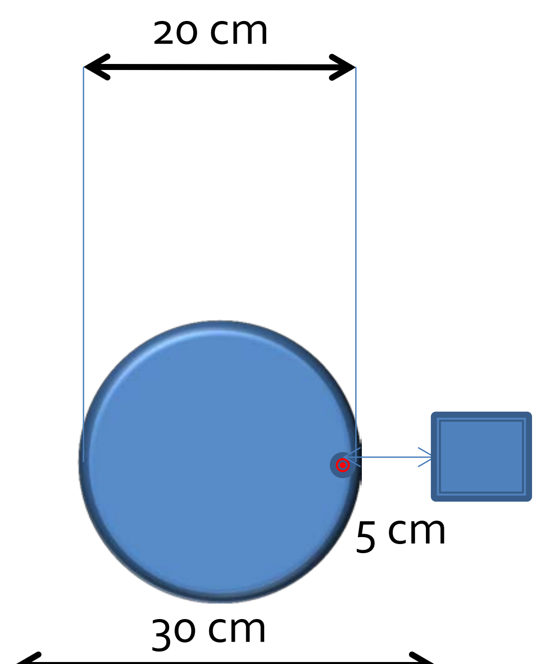
For a fixed obstacle distance of 30 cm

Different Angle Mounting in Deg	Measured Distance in cm
90	30
80	31
70	33
60	31
50	32
20	31

Mounting of IR Sensor

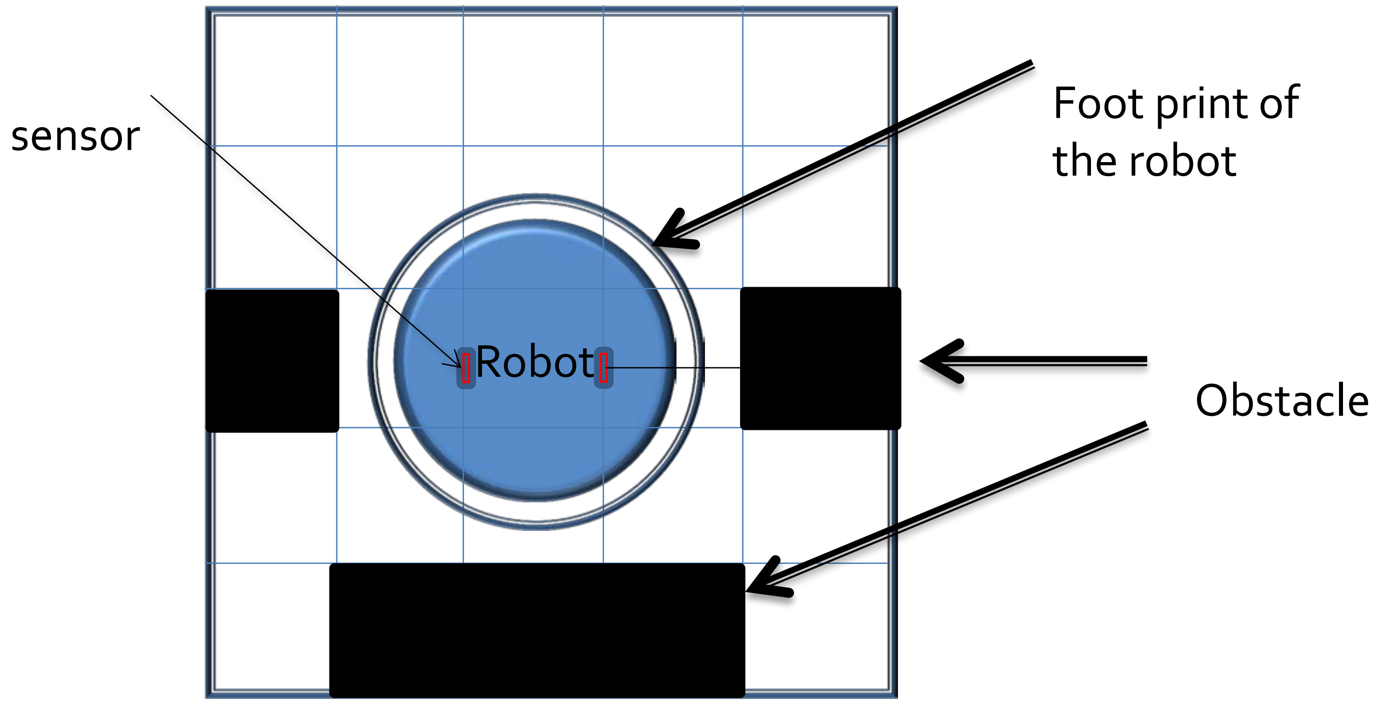
Fig.5 Analog Output Voltage vs. Distance to Reflective Object





Placement of sensor will increase the robot footprint

Mounting of IR Sensor



Understanding the Motor and Encoder

Source: https://www.pololu.com/product/2285

Motor:

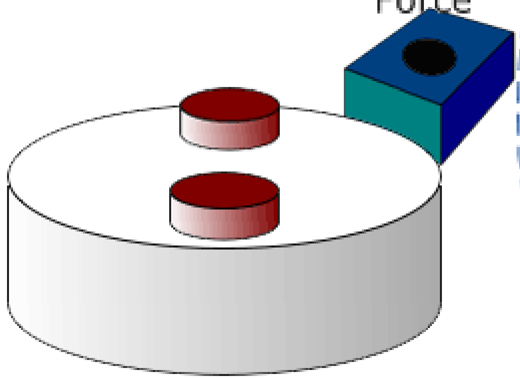
- Low power **6V** DC-brushed motor
- **46.85:1** metal spur gearbox
- 120 RPM and 250 mA free-run, 65 oz-in (4.7 kg-cm) and 2.4 A stall

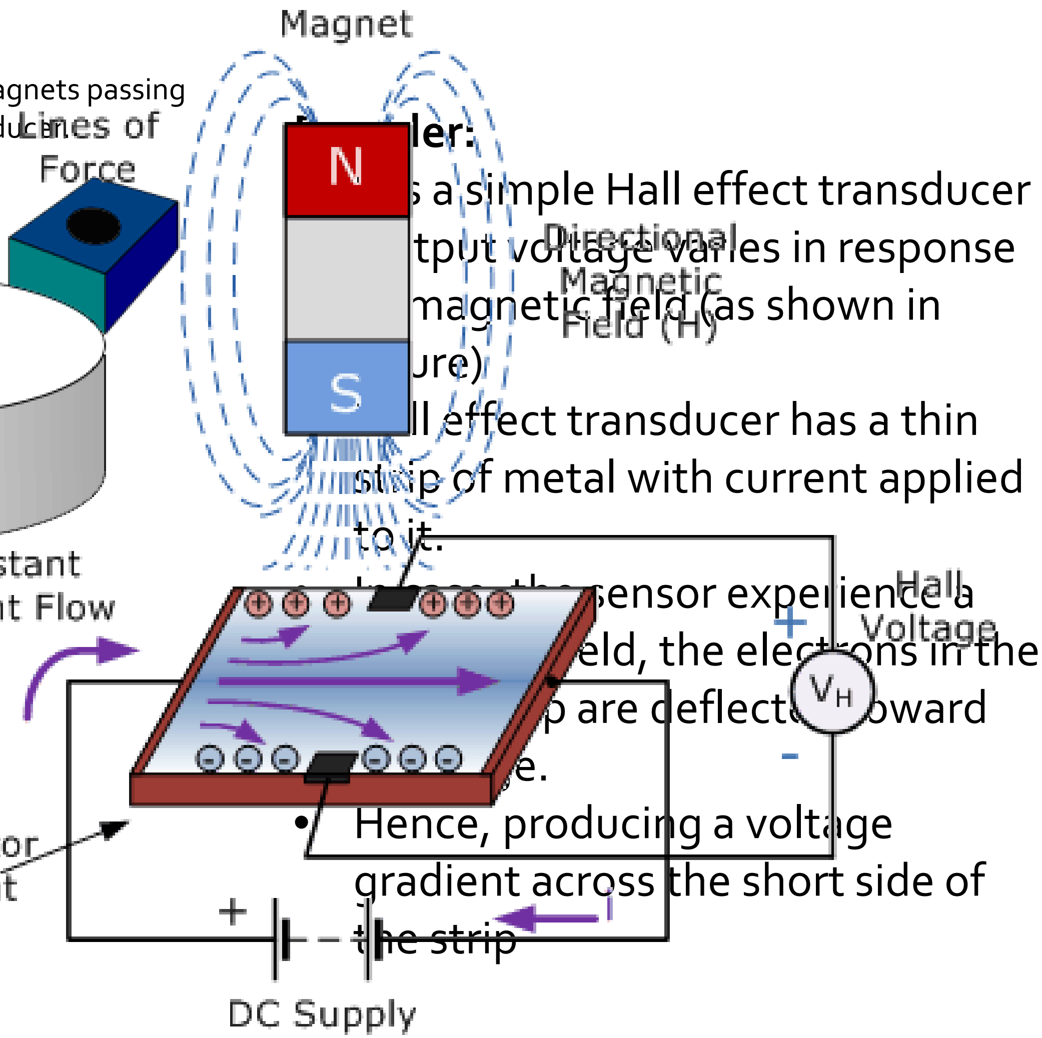
Encoder:

- 3.5V and draws a maximum of 10 mA
- integrated 48 CPR quadrature encoder
- No of counts per revolution
  - 46.85 times 48 = 2248.86 CPR

Understanding the Encoder

Wheel containing two magnets passing hall effect transducer.





...ler:

- ...s a simple Hall effect transducer ...tput voltage varies in response ... magnetic field (as shown in ...ure)
- ...ll effect transducer has a thin strip of metal with current applied to it.
- ... sensor experience a ...eld, the electrons in the ...p are deflect... toward ...e.
- Hence, producing a voltage gradient across the short side of the strip

Understanding the Encoder

Overview

Open Loop Control

PID Control

2 WD Speed Control

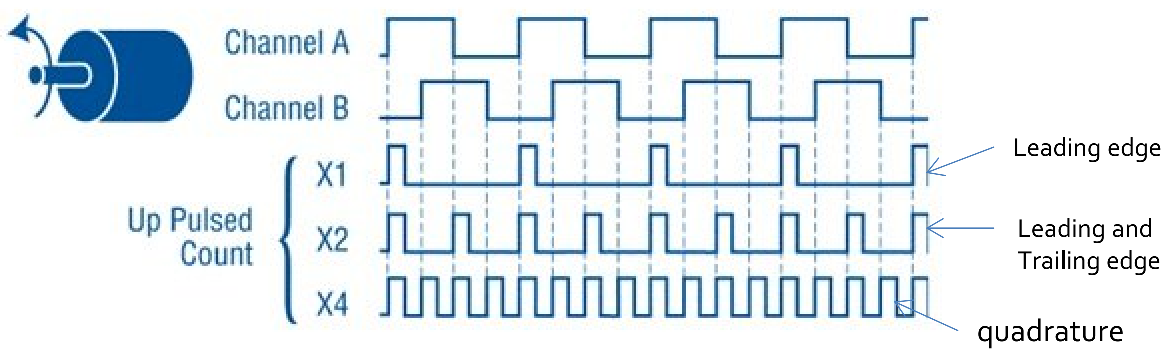
Issues

- Number of pulses per revolution is 562 pulse per revolution
- Find time taken by one pulse using 'pulsein' function and compute RPM
- 'pulsein' function Arduino works perfectly for finding pulse-width between 10 microseconds to 2 minutes
- Another way is to use the timer1 input capture mode. You can do background computations to speedup. Refer to the thread: http://www.arduino.cc/cgi-bin/yabb2/YaBB.pl?num=1201890734
- Don't use 'millis' function in ardunio. If you use then error could be accumulated.

Understanding the Motor and Encoder

Encoder:

- Normally, the output signal from encoder is at least one square wave, but most often it sends two square waves with their phase shifted at 90 degrees – known as quadrature.
- Resolution
  - PPR – Pulse per Revolution
  - CPR – Count per Revolution



Source: https://www.dynapar.com/technology/encoder_basics/quadrature_encoder/

Overview

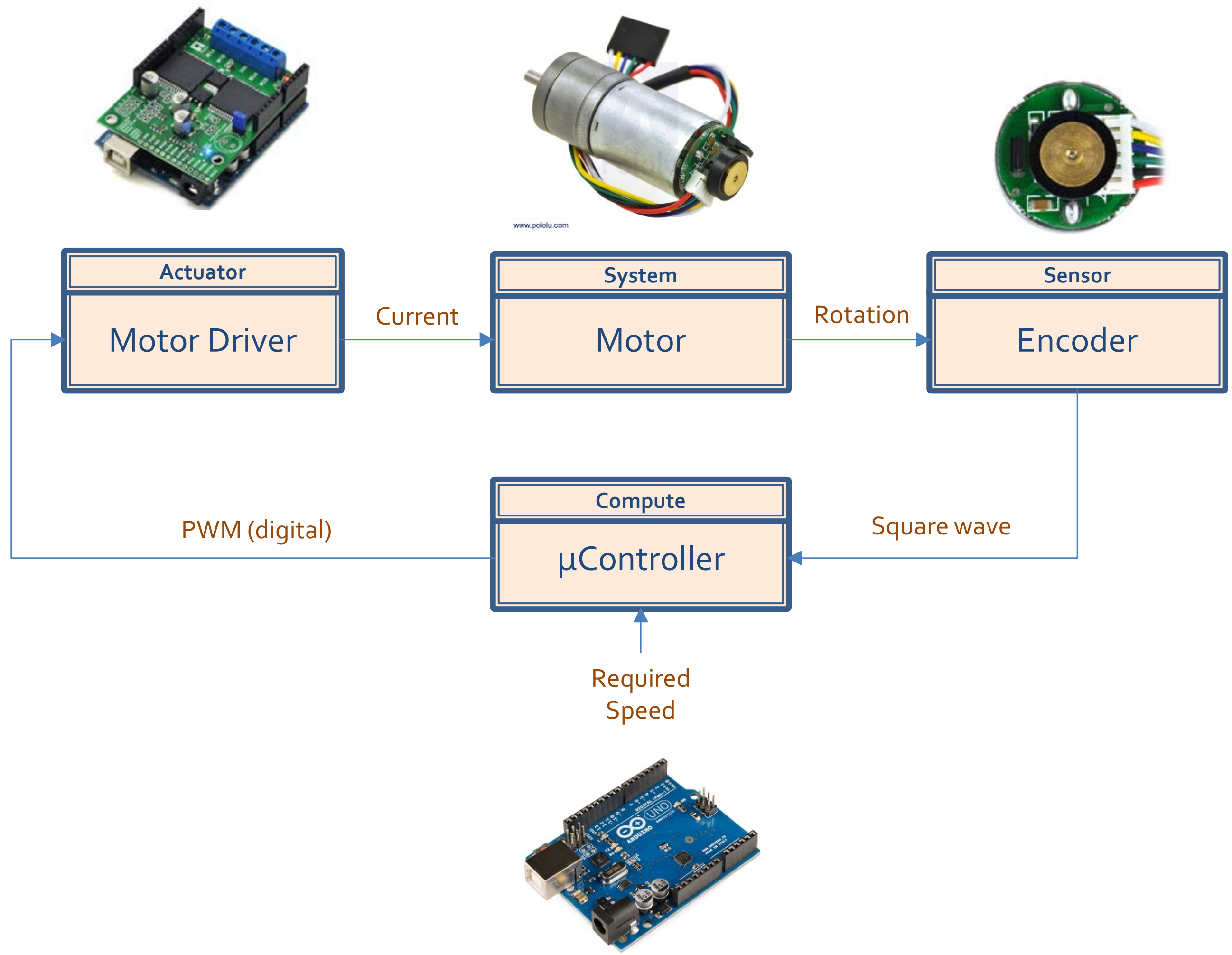
Open Loop Control

PID Control

2 WD Speed Control

Issues

Overview of Speed Control System



Overview

Open Loop Control

PID Control

2 WD Speed Control

Issues

Understanding of Actuator

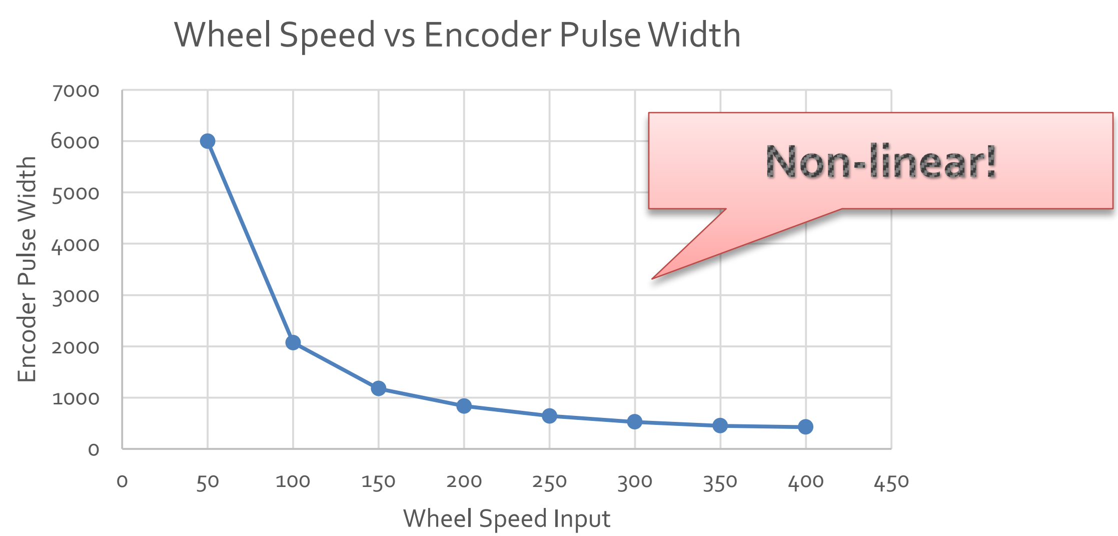
- Microcontroller board drives the motor through PWM.
- In the board library, required Motor Speed is specified by (u) and is in the range of -400 to +400.
- Driver sends corresponding current to motor to drive it
- Overall relationship -

  *subject to battery charge, surface friction, etc

Speed	RPM
400	131.863
350	122.2386
300	103.6323
250	84.4372
200	65.01241
150	46.05117
100	27.73403
50	9.67545
0	0

Understanding the Sensor

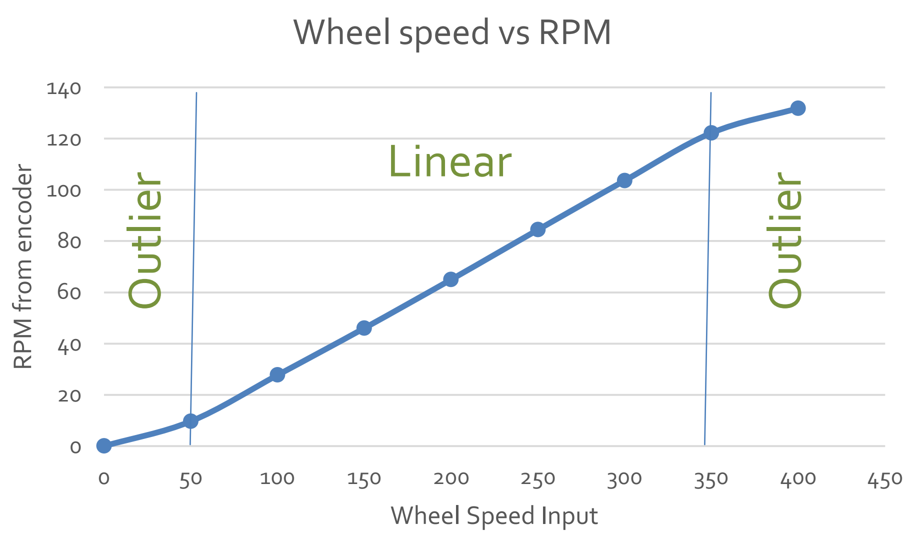
- Encoder is used to measure the speed of motor
- Need to convert square wave from motor encoder to a meaningful speed!!
- Using time-width of pulse is one way -> faster the wheel speed, longer the time-width :



- Overview
- Open Loop Control
- PID Control
- 2 WD Speed Control
- Issues

Understanding the Sensor

- Convert time-width to rpm of wheel -> (note 1124.5 square waves for every revolution of wheel)



Open Loop Control

- Assume: perfect system. We can achieve straight line motion using table look-up!
- Setting equal speed on both wheel!!!

Overview

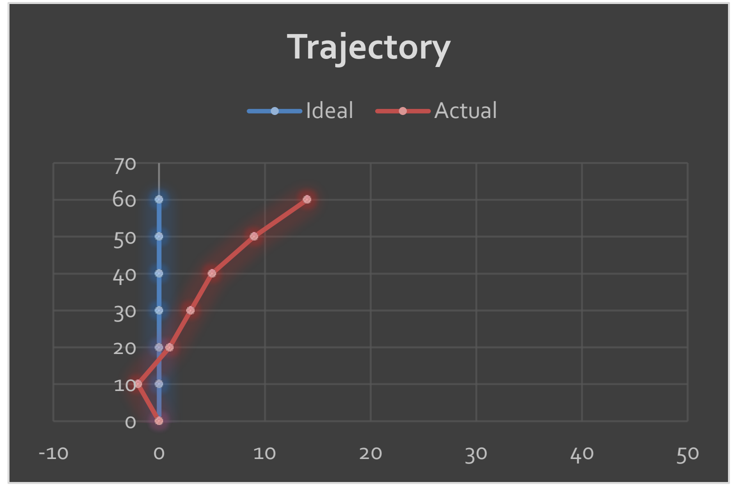
Open Loop Control

PID Control

2 WD Speed Control

Issues

Open Loop Control

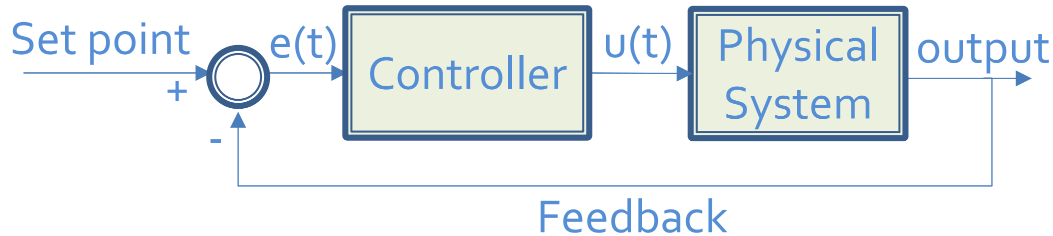


- Open loop control system will not work due to various reasons!

1. Motor characteristics are different
2. Friction of surface
3. Uncertainties
4. PWM calculation errors

Closed-Loop Control

Basic PID Control System



- Physical System – Motor
- Input u(t) – Motor current (0-12 A)
- Output – Motor speed (0-130 rpm)
- Set point – Required rpm
- Error e(t) = (Set point – Feedback)
- **PID Controller tries to minimize the error and handle uncertainties.**

PID Control

PID – Proportional, Integral, Derivative Control

- PID is one type of control law!
- Analog PID Implementation:
  - $u(t) = K_P * \mathrm{e(t)} + K_I * \int_0^t e(\tau)d\tau + K_D * \frac{de(t)}{dt}$

Reduce the current error

Reduce a steady offset from set point

Reduce the increase/ decrease in error

Digital PID

- Digital PID Implementation*:

$$u[k] = u[k-1] + K_1 * e[k] + K_2 * e[k-1] + K_3 * e[k-2]$$

Where

$K_1 = K_p + K_i + K_d$

$K_2 = -K_p - 2K_d$

$K_3 = K_d$

*K.J. Astrom and T.Hagglund. PID Controllers, 2nd ed., Instrument Society of America, 1995.

- Difference equation -> No need to integrate/differentiate in the code!
- The method to find K1,K2,K3 will be discussed later!!

Digital vs Analog

Comparing Digital and Analog Controllers

Digital PID in micro controller	Analog PID
More economical because of cheap components and the simple design algorithm	Comparatively expensive due to the complexity of the design algorithm
Fully integrated and compact	A large number of operational amplifiers and other components are needed
High noise immunity	Noise susceptibility is high
More flexibility because of the ability to program and reprogram the chip	Redesigning is required for any change in the system parameters
High accuracy with faster processing and low power consumption	Less accurate with more processing time and power consumption is higher

Abdul Rasool et al (2009), Sudan Engineering Society Journal

Overview

Open Loop Control

PID Control

2 WD Speed Control

Issues

Pseudo code for Digital PID

Algorithm to implement PID control in any micro processor

1. Fix kp,ki,kd
2. Calculate k1,k2,k3
3. Read feedback signal into 'y' (Data Acquisition)
4. Calculate error between set point and feedback
5. Compute 'u' using digital PID control law
6. Send 'u' to output for control
7. **Wait for 'xx' msec (sampling time) before continuing**
8. Repeat 3-7

Other considerations – (a) Need to know kp,ki,kd

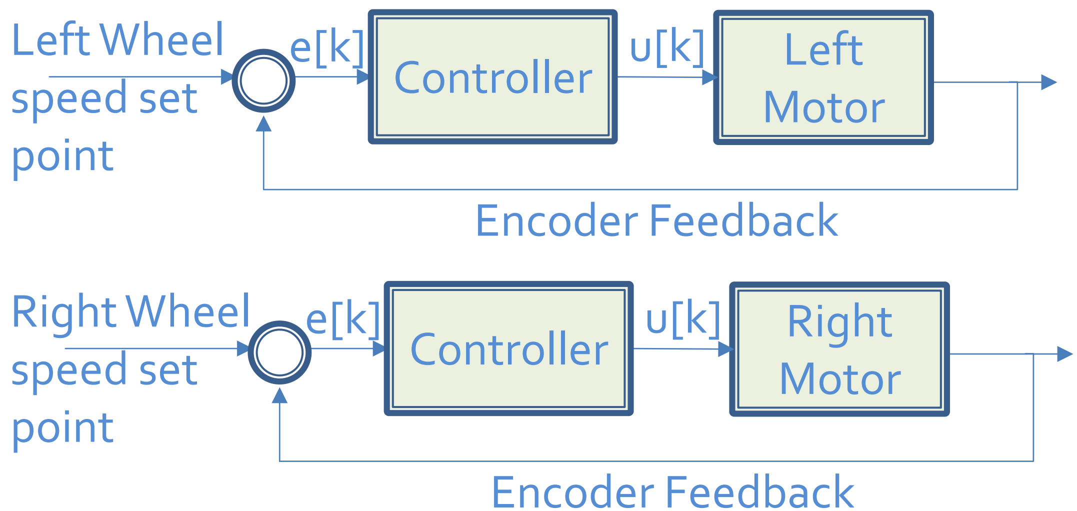
(b) Sampling Time

(c) Input/output saturation, etc.

-> Will be covered in the slides to come

2-wheel drive PID control

2-wheel drive car



- 2 independent control loops
  - Micro-controller implements the two above control loops

2-wheel drive PID control

- Control Loop:
  - <u>Set point</u> – Determined by scenario, eg:
    - Go-forward => left and right wheel SP = ‘xx’ rpm)
    - Turn-left -> left wheel SP = ‘-xx’ rpm, right = ‘xx’ rpm for a particular duration
  - <u>Controller</u> –
    - Need to determine tuning parameters to achieve the set point
  - <u>Output</u> –
    - Need to convert the controller output to PWM – Motor driver library can be used
  - <u>Feedback</u> –
    - Need to convert the encoder feedback to speed – needs understanding of the encoder specification

Tuning Parameters

Next logical question – How to choose the K_p, K_i, and K_d ?

- Various Methods :
  - *Ziegler-Nichol's Open Loop Tuning Method –*
    Note: $K_p = K_c$, $K_i = K_c/T_i$, $K_d = K_c * T_d$

Controller	Parameters		
	K_c	T_i	T_d
P	$\frac{\tau}{K\theta}$		
PI	$\frac{0.9\tau}{K\theta}$	$\frac{\theta}{0.3}$	
PID	$\frac{1.2\tau}{K\theta}$	2θ	0.5θ

Seborg, et al, Process Dynamics and Control, Second Edition..

Overview

Open Loop Control

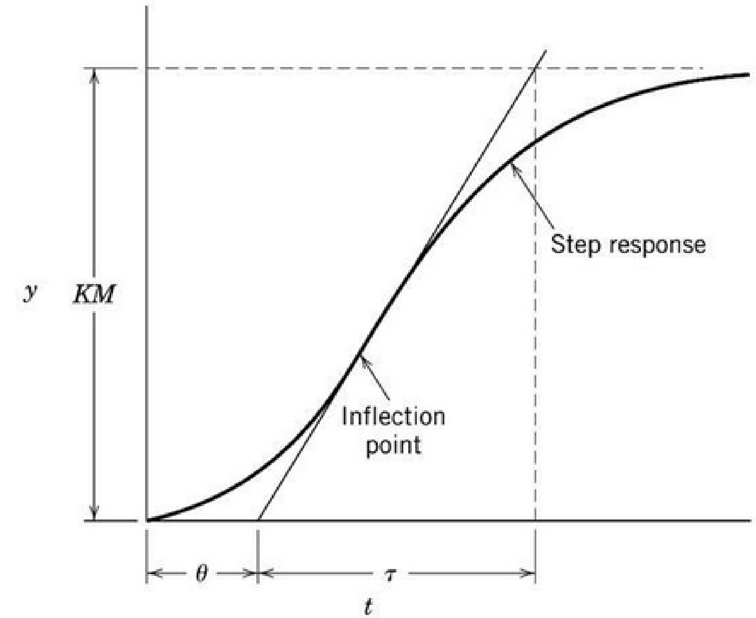
PID Control

2 WD Speed Control

Issues

Tuning Parameters

- Step Test to determine K, τ and θ

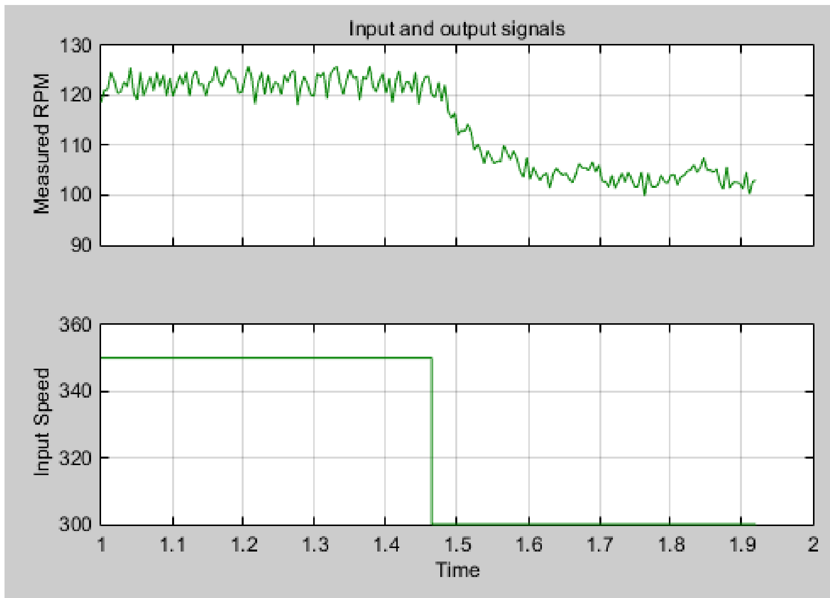


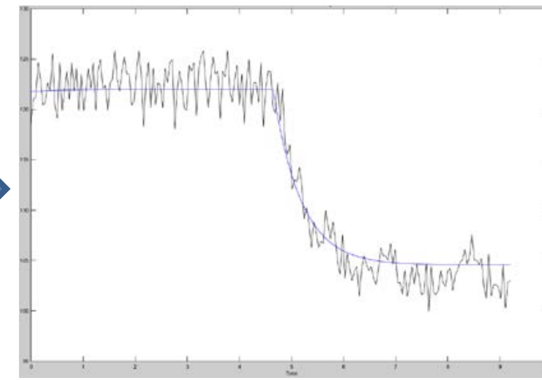
Response of output to an input change of 'M' at t = 0 sec

- Determine τ and θ
- Determine K = KM/M

Tuning Parameters

- Actual Test data from motor step test –





Determine K, τ and θ

Determine K_p, K_i, K_d

Overview

Open Loop Control

PID Control

2 WD Speed Control

Issues

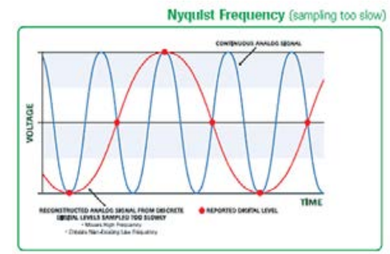
Closed-loop control

Implementation Issues

- **Excessive noise** in input signals
  - Consider filtering
- **Slow processor** – sluggish control
  - Ensure all computations are complete within sampling period
- **Saturation** – breaching limits
  - Impose validity limits for inputs, calculations, outputs.
- **Slipping –** during frequent start/stop
  - Avoid jerky control

Sampling Rate

Rate of data acquisition from feedback -> Important!



- **Too slow**
  - Controller acts slowly, feedback sampling violates nyquist criterion
- **Too fast**
  - Excess noise in differentiator, overflow in integrator
- **Ideal**
  - Rule of thumb -> Sampling Rate =
    (0.01 to 0.1)*time constant of system

Beyond this …

- We have discussed speed control of 2 wheel drive robot
- Other tasks in navigating maze -> obstacle avoidance, exploration will be taken care by top-level mission control

Good Luck!